

人体检测及行为分析的研究 技术笔记

王官皓

Abstract—人体检测及行为分析的研究中涉及到的技术细节以及毕业设计的诸多细节。

Index Terms—行人检测, 行为分析, 技术笔记.

I. 级联器训练 (CASCADE CLASSIFIER TRAINING)

本节记录综述 [1] 中的 Haar 级联器用 OpenCV 运算库进行实现的细节, 主要涉及到训练数据的准备 (样本的生成), 训练, 分类的实现. 参考了 [2], [3], [4], [5] 等资料.

OpenCV 提供了两种训练方法: `opencv_haartraining` 和 `opencv_traincascade`. 后者是较新的版本, 在 OpenCV 2.x API 框架下采用 C++ 实现, 故采用后者. `opencv_traincascade` 可以采用 TBB 库进行多线程运算, 需要用 TBB 编译的 OpenCV 库. 训练之前可用 `opencv_createsamples` 来生成阳性样本和训练样本, 输出格式为 `*.vec` 格式, 是包含图像数据的二进制格式.

A. 训练数据准备

训练样本有两种类型: 阴性样本和阳性样本. 阴性样本没有包含目标对象, 阳性样本则包含了待检测的对象. 阴性样本必须手工准备, 而阳性样本可以采用 `opencv_createsamples` 自动生成.

1) 阴性样本: 阴性样本可以从任意不包含待检测对象的图像中采样, 阴性样本需要以特定格式列举在一个描述性的文本文件中, 每一行包含一个文件名, 需要注意的是样本中的图像分辨率需要大于训练窗口尺寸. 描述文件的示例如下:

目录结构 (阴性样本放置于 `negative_images` 文件夹内):

```
1 /negative_images
2   img1.pgm
3   img2.pgm
4 negatives.txt
```

生成的文件列表描述文件 `negatives.txt` 格式:

```
1 negative_images/img1.pgm
2 negative_images/img2.pgm
```

文档中要求手动生成, 然而可以采用 `bash` 的 `find` 命令来自动生成文件列表描述文件:

```
1 find ./negative_images -iname "*.pgm" > negatives.txt
```

2) 阳性样本: 阳性样本通过 `opencv_createsamples` 来生成, 可从单一图像或是经过预标记的图像文件中提取. 阳性样本的数量依赖于特定应用, 例如, 在识别公司 logo 的应用中, 可能只需要 1 个阳性样本, 而在人脸识别或是人体识别中, 需要数以千计甚至更多的样本. 关于 `opencv_createsamples` 的参数说明:

- `-vec <vec_file_name>`: 输出文件名
- `-img <image_file_name>`: 源文件名
- `-bg <background_file_name>`: 背景描述文件, 用于对象随机失真背景
- `-num <number_of_samples>`: 生成的阳性样本数量
- `-bgcolor <background_color>`: 背景颜色 (透明), 可以和 `-bgthresh` 配合设置背景色彩容限, 在 `bgcolor-bgthresh` 和 `bgcolor+bgthresh` 区间内的像素视作透明.
- `-inv`: 设置反色
- `-randinv`: 随机反色
- `-maxidev <max_intensity_deviation>`: 前景样本内像素的最大强度偏差
- `-maxx(y/z)angle <max_x(y/z)_rotation_angle>`: 最大旋转角度
- `-show`: 调试选项, 可以显示样本
- `-w <sample_width>`: 输出样本的宽度
- `-h <sample_height>`: 输出样本的高度

源图像会根据参数设置随机旋转, 获得的图像随机放置在背景描述文件指定的任意背景上, 按照参数设置的尺寸保存在 `*.vec` 文件中. 阳性样本也可以从预标记的图像集合内获取, 图像集合需要一个描述性的文本文件, 每一行描述一个文件, 以文件名开始, 后面接对象数量和对对象坐标 (`(x,y,width,height)` 格式). 描述文件的示例如下:

目录结构:

```
1 /positive_images
2   img1.pgm
3   img2.pgm
4 positives.txt
```

生成的列表描述文件 `positives.txt` 文件格式:

```
1 /positives_images/img1.pgm 1 140 100 45 45
2 /positives_images/img2.pgm 2 100 200 50 50 50 30 25 25
```

从以上阳性样本集合中创建样本, 需要 `-info` 参数:

- `-info <collection_file_name>`: 描述文件名

不用设置失真, 所以只还需要 `-w`, `-h`, `-show`, `-num` 等参数.

Daimler 公司提供的数据集内的阳性样本 (DaimlerBenchmark/Data/TrainingData/Pedestrians) 都是经过预标记的, 所以只需要使用 `bash` 命令 `find` 可以生成阳性样本描述文件 (以 18×16 分辨率为例):

```
1 find ./positive_images/ -name '*.pgm' -exec \
2   echo \{\} 1 0 0 18 36 \; > positives.txt
```

然后可以进行样本创建:

```
1 opencv_createsamples -info positives.txt\
2 -vec positives.vec -w 18 -h 36
```

创建完成后可以使用-show参数进行查看:

```
1 opencv_createsamples -vec positives.vec -w 18 -h 36
```

B. 级联器训练

经过前面训练数据集的预处理准备, 接下来采用 `opencv_traincascade` 来得到期望的级联器. 在训练完成后, 级联器会保存在 *.xml 文件中. 关于 `opencv_traincascade` 的参数:

- `-data <cascade_dir_name>`: 级联器保存参数
- `-vec <vec_file_name>`: 前面得到的阳性样本文件名
- `-bg <background_file_name>`: 背景文件 (阴性)
- `-numPos(Neg) <number_of_positive(negative)_samples>`: 级联器每一层采用的阳性/阴性样本的数量
- `-numStages <number_of_stages>`: 级联器级数
- `-precalcValBufSize <vals_buffer_size>`: 预处理特征值的缓存区大小 (Mb)
- `-precalcIdxBufSize <idxs_buffer_size>`: 预处理特征值索引的缓存区大小 (Mb), 与训练速度正相关.
- `-baseFormatSave`: 文件格式选择, 指定后会存为旧格式
- `-stageType <BOOST(default)>`: 层类型
- `-featureType <HAAR(default),LBP>`: 特征类型, HAAR-Haar 特征, LBP [6]-局部二值特征¹.
- `-w(h) <sampleWidth(Height)>`: 训练样本的尺寸, 必须与样本生成中采用的尺寸一致.
- `-bt <DAB,RAB,LB,GAB(default)>`: 级联类型: DAB-离散 AdaBoost, RAB-Real AdaBoost, LB-LogitBoost, GAB-Gentle AdaBoost.
- `-minHitRate <min_hit_rate>`: 单级检测率要求, 整体检测率大概为 $\min_hit_rate^{number_of_stages}$.
- `-maxFalseAlarmRate <max_false_alarm_rate>`: 最大误判率要求, 整体误判率大概为 $\max_false_alarm_rate^{number_of_stages}$.
- `-weightTrimRate <weight_trim_rate>`: 指定剪枝及权重, 建议选择为 0.95.
- `-maxDepth <max_depth_of_weak_tree>`: 树的最大深度, 建议选择为 1.
- `-maxWeakCount <max_weak_tree_count>`: 单级树数量, 为了满足 `-maxFalseAlarmRate` 参数要求单级需要有 $\leq \maxWeakCount$ 个树.

¹虽然 [1] 中的结论指出 Haar 级联器在低分辨率和实时处理条件下表现最优, LBP 特征与 Haar 特征相比训练和检测还会快许多倍, 而分类的质量高度依赖于训练数据集和训练参数, 训练出和 Haar 级联器质量相同的 LBP 级联器是可能的.

- `-mode <BASIC(default)|CORE|ALL>`: 选择 Haar 特征类型. BASIC-采用垂直特征, ALL-采用所有特征 (垂直和旋转, 如综述 [1] 中所示).

[1] 中指出级联层数 N_l 在 $N_l = 15$ 时达到饱和, 按照其参数选择, 在 18×36 阳性样本分辨率下, 配置 15 层级联, 采用所有 Haar 特征, 单层在 15660 个阳性样本和 15660 个阴性样本下训练, 选定单级 50% 的误判率和 99.5% 的检测率², 运行时特征值缓存区和特征值索引缓存区大小设置为 1024MB 和 1024MB, 命令如下:

```
1 opencv_traincascade -data classifier -vec positives.vec\
2 -bg negatives.txt -numStages 15 -minHitRate 0.995\
3 -maxFalseAlarmRate 0.5 -numPos 15660 -numNeg 15660\
4 -w 18 -h 36 -mode ALL -precalcValBufSize 1024\
5 -precalcIdxBufSize 1024
```

无论是社区还是实际操作来看, 训练数据的准备是比较快的 (只涉及到转换为二进制文件, 计算消耗不大), 但是训练过程非常缓慢, 社区文档显示 2011 年产的 Macbook Air 在 10^3 数量级的样本数量条件下会耗时一周左右, 样本量在 15660 的条件下可想而知训练时间会几何级地上升, 采用普通计算机是不可行的. 采用 AWS EC2 来操作或许是一种可行的方法.

C. 分类

在训练完成后, 利用获得的级联器 *.xml 文件, 可以进行分类测试.

2014-3-1, 下一步: 实现测试程序, 从测试图像中读取并进行检测, 对检测结果加边框标记.

REFERENCES

- [1] Enzweiler, M.; Gavrilu, D.M., "Monocular Pedestrian Detection: Survey and Experiments," Pattern Analysis and Machine Intelligence, IEEE Transactions on , vol.31, no.12, pp.2179,2195, Dec. 2009 doi: 10.1109/TPAMI.2008.260
- [2] Cascade Classifier Training, http://docs.opencv.org/doc/user_guide/ug_traincascade.html
- [3] Cascade Classification, http://docs.opencv.org/modules_objdetect/doc/cascade_classification.html
- [4] Learn how to train your own OpenCV Haar classifier, <https://github.com/mrnugget/opencv-haar-classifier-training>
- [5] Tutorial:OpenCV haartraining, <http://note.sonots.com/SciSoftware/haartraining.html>
- [6] Shengcai Liao, Xiangxin Zhu, Zhen Lei, Lun Zhang and Stan Z. Li. Learning Multi-scale Block Local Binary Patterns for Face Recognition. International Conference on Biometrics (ICB), 2007, pp. 828-837.

²按照文档给定的估计方法, 整个 15 级系统的检测率为 $0.995^{15} = 0.9276$, 是比较低的, 社区内的代码建议为单级 0.9999.